SNMP Technical Introduction, Part 3 of 3

Revised: 11/24/92
Security: Everyone

SNMP Technical Introduction, Part 3 of 3

==========

Article Created: 10 November 1992

TOPIC ----------

This is the third of three articles that introduce SNMP (Simple Network Management Protocol). This article describes SNMP Protocol and Operations, Management Concepts, and concludes with a comparison of OSI vs. SNMP.

DISCUSSION ----------

SNMP Protocol and Operations

The next step is to get the management console(s) in communication with the nodes on the network.

Messages
Communication among protocol entities is accomplished by the exchange of messages, each of which is entirely and independently represented in a datagram. SNMP messages require only an unreliable datagram service, and every message is entirely and independently represented by a single transport datagram.

With SNMP, TCP/IP networks use User Data Protocol (UDP) to exchange messages. With AppleTalk, the Datagram Delivery Protocol (DDP) is used.

A message consists of a version identifier, an SNMP community name, and a Protocol Data Unit (or PDU), which contains the command and its operands. As was the case with MIBs, messages and data all use the basic encoding rules of OSI's Abstract Syntax Notation One (ASN.1). This allows data exchanges to occur smoothly over a variety of platforms.

SNMP Command Set
The network management protocol allows the variables of an agent's Management Information Base (MIB) to be inspected or altered. The protocol is based on sending requests and getting responses. Most commands are

requests to either set the value of some parameter or to retrieve such a value.

To make SNMP simple, its designers defined a limited command set. A console can issue only three commands: Get, GetNext, and Set.

- The Get command allows a console to read a specified management variable in the managed device. An agent would execute a GetResponse command to return desired values to the console.

- The GetNext command allows the console to read the subsequent variables in the agent's database (MIB) without specifying the name of that particular variable. This command allows even consoles that do not have the appropriate MIB installed to manage SNMP based network devices. Perhaps the most powerful use of this command is to obtain an entire string of data with one call, as GetNext accepts multiple operands. SNMP advocates like to refer to this command as the "powerful GetNext command."

A console can also modify certain variables for which the console has the required access privileges using the Set command. This allows an administrator to reset a variable that keeps count of a certain occurrence, or to remotely set a flag that signals an agent to perform some specified function. An agent would execute a GetResponse command to return desired values to the console.

An agent can issue two commands: GetResponse and Trap.

- The GetResponse command is used by an agent to assemble and return values to the console.

- Finally, the Trap command allows a managed entity to send a message to a specific console, when a specified event occurs. The agent is responsible for detecting the event and issuing the Trap command.

Trap Implementations
Traps must be pre-programmed into Network Elements. They are not down line loaded to the network elements by the console.

SNMP uses a method called trap-directed polling. When an event causing a trap occurs, the managed node sends a single simple trap to the console. The console is responsible for initiating further interactions with the managed device to determine the problem. This is quite different from a polling approach, where each network element is constantly probed for status. It also minimizes management traffic on the network.

SNMP Management Concepts

SNMP Community
In the SNMP environment, a community is a relationship between SNMP agents and SNMP managers.

Each SNMP community is named by an NVT ASCII text string that is called the community name.

Community Views

Each managed device contains a community view. This view determines how much access (read only, read and write, or no access at all) the management console has for each given object.

Managed devices can belong to several communities at once. Each console can therefore have different views of the data contained in the managed device.

Security

The community name is a security feature in SNMP. For a console to reference a MIB or issue a set command, it must have the correct community name. Unfortunately, community names are not encrypted. Community names use the ASN.1 Display String (text) object type, which is transmitted in NVT ASCII. Someone monitoring the network with a packet analyzer could ascertain the community name in use by a particular console. IETF is currently working on a new security scheme for SNMP, which would encrypt community names.

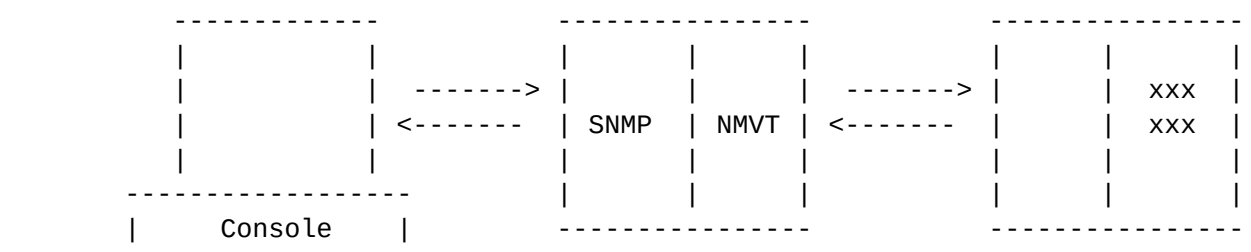
If the community name matches a name configured in the managed device, the console must have a correct view. As discussed earlier, this view determines if the console has read access, read and write access, or no access at all.

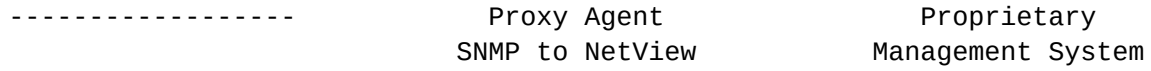
Managing "Foreign" Devices Using Proxy Agents

SNMP is independent of the architecture and mechanisms of particular hosts or particular gateways. SNMP can manage devices that do not conform to the architecture using Proxy Agents. Proxy agents allow SNMP to be used in environments that comprise multi-vendor devices using different protocols and different management schemes.

In the case of nodes that do not understand SNMP, a Proxy Agent can be used to translate one management protocol to another. An example would be a Proxy Agent that converts SNMP into Network Management Vector Transport (NMVT), the protocol used by IBM's NetView.

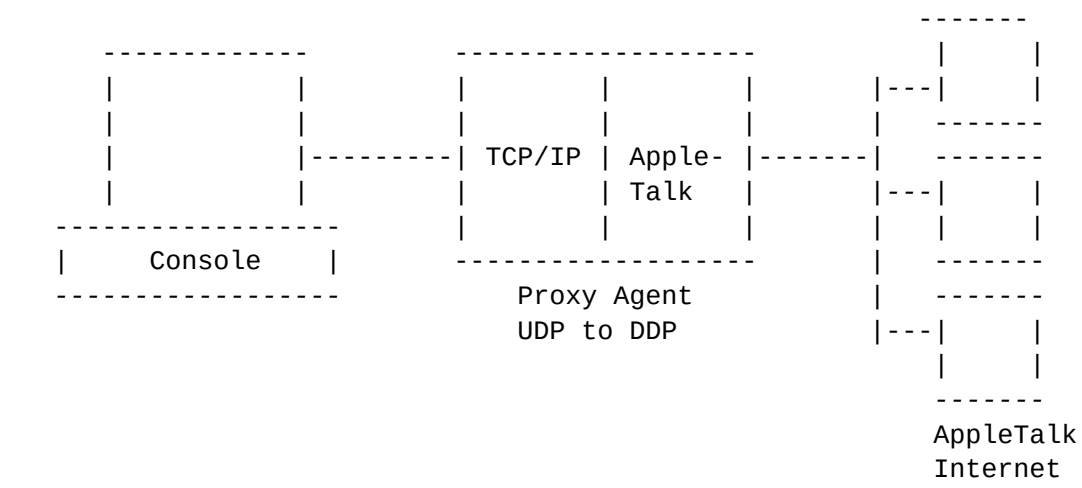
In the example below, the Proxy Agent is performing the aforementioned translation. An example is IBM's NetView 6000 running on an RS6000 platform, which implements SNMP translation for SNA networks. It is acting as an "application gateway."





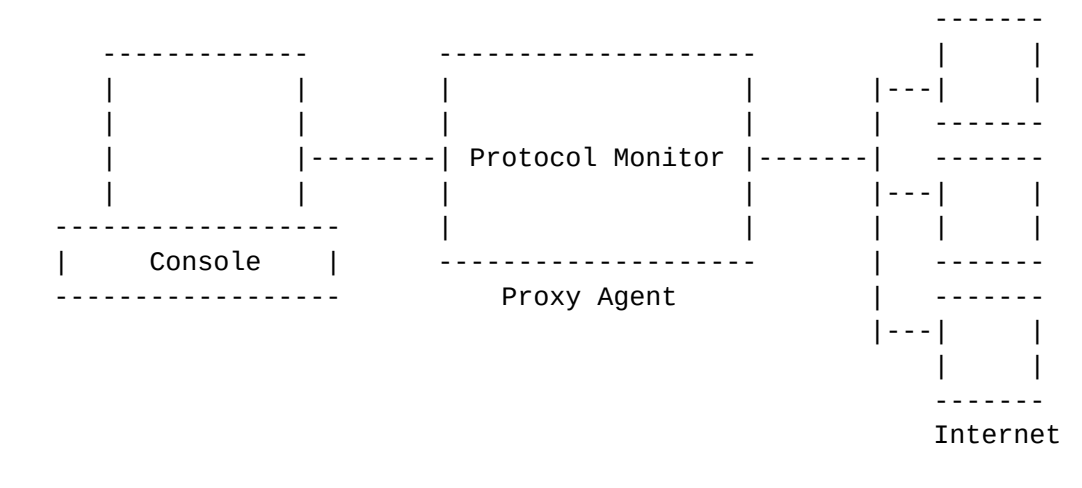
In the case of nodes that do not understand TCP/IP, but do support SNMP, a Proxy Agent can strip out the SNMP-specific data units and encapsulate them in another transport. A more complicated function of this type of agent is address remapping/resolution between the two networks.

Below is an illustration of a Proxy Agent that allows consoles using TCP/IP to manage AppleTalk networks using MacSNMP. This agent is acting as a "transport gateway" in standards parlance. Examples of this are the Cayman GatorBox, the Shiva FastPath, the Compatible System EtherRoute, and Cicso routers running release 9.0 gateway software.

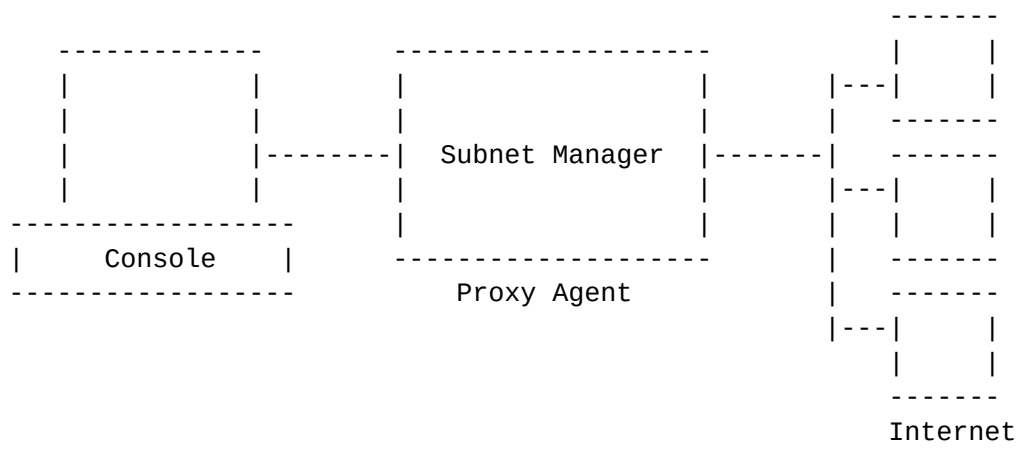


Other types of Proxy Agents can be used in SNMP networks. For example, Proxy Agents can sit on a network and monitor packet traffic, gather statistics, and send traps to consoles when certain conditions occur.

This is illustrated below. The console could then query these Proxy Agents to obtain their statistics, or to further analyze the trap condition reported. Examples are LANtern from Novell, Asante's network management hubs, and protocol monitor from Network General that also captures and decodes packets.



A Proxy Agent can be responsible for specific nodes on the network, monitoring them on behalf of the console and sending a high-level trap when something of interest happens. This is illustrated below.



This Proxy Agent has the added advantage of cutting down on network traffic, and frees the console from having to monitor a large number of nodes on a one-to-one basis. Subnet managers are under development by several vendors.

OSI vs. SNMP

SNMP and CMIP Differences

Many people are confused about the differences between SNMP and OSI's Common Management Information Services (CMIS) and its associated protocol, Common Management Information Protocol (CMIP).

Part of the confusion is that both protocols share some things in common. Both use Management Console/Agent architecture, and both use MIB structures to keep track of information -- but this is where the similarities end.

SNMP is a connectionless oriented protocol. Its commands and responses are transported with datagram transports like UDP and AppleTalk's DDP. These transports do not require formal session establishment between the manager and the managed to pass commands or responses. CMIP is a connection oriented protocol, requiring a session to be established between the manager and the managed to pass information. In the TCP/IP world, CMIP would use TCP as transport.

Another difference is the command set. CMIP uses Get and Set commands like SNMP, and has a command (similar to the SNMP Trap) called Event Report. The difference in the command set is that CMIP implements three additional commands: action, which can command a device to reboot for example; create, which is used to create new objects; and delete, which deletes objects.

One can duplicate the functions of these three commands in the SNMP world using the Set command to change a variable, which would then trigger an agent to perform specific programmed actions duplicating the function of action, create, and delete, but the SNMP command set does not support these

functions directly.

Additionally, CMIP uses a concept called scoping to locate a particular value by passing its locations and depth in the target node's tree. Further, these values can be filtered to segregate information.

What Is CMOT?

CMOT, short for CMIP Over TCP, implements the CMIP protocol in TCP/IP networks. Note the use of the connection oriented protocol, TCP, instead of UDP datagrams.

In Conclusion

The simplicity of the SNMP architecture and command structure has been discussed. It is a proven, established direction, with over 50 companies offering more than 200 networking products that support SNMP. SNMP is here today.

SNMP can be enhanced by extending the MIB to support product-specific features of different manufacture. This structure must be imported and compiled by the console to intelligently access different portions of a vendor-specific MIB.

Although SNMP seems simple on the surface, the detail is really in the MIB portion of the implementation, different than other management protocols with large detailed command sets.

SNMP has received some criticism in some areas. For instance, there is no capability defined to ensure that SNMP commands received by an agent really originated from an actual authorized management console. The IETF is working hard on improving security in SNMP, and this problem will be resolved shortly.

In the meantime, MIB variables can be protected by making them read only, but this will not allow the use of a Set command to change them. This lack of sophistication in authorization has made vendors reluctant to offer extensive functions that take advantage of SNMP's Set command. Hence, most routers are still configured with the vendor's own management utilities.

Additionally, the Trap command has caused practical problems in its implementation, as a managed device must constantly evaluate its function to determine if a trap should be executed, spending valuable CPU time and occupying memory in the process. In practice, this impact has proved to be relatively minor.

SNMP uses datagrams for transport. Datagrams do not provide delivery assurance. Messages can arrive out of sequence, or may be lost or duplicated. Most vendors implement a numbering scheme to address out-of-sequence conditions and duplications -- but messages can still get lost. On the other hand, when a network is in trouble, how can it be expected to support connection oriented sessions when the physical transport is unreliable?

In comparison to other network management proposals such as OSI's CMIP, SNMP is viewed by the OSI community to be not as versatile or applicable to a broad range of network products. In fact, some view SNMP as a "down-and-dirty fix" until CMIP is ready. And although SNMP's authors have done their best to map SNMP architecture into CMIP, they are incompatible.

However, SNMP's public domain distribution, relative ease of implementation compared to other network management scenarios, and its growing installed base make it a popular choice for users with multi-vendor networks.

Copyright 1992, Apple Computer, Inc.

Tech Info Library Article Number:11017